LanguageLeap Release 1 Summary

Team Members

Name	Email
Alan Ly	hello@alan.ly
Thomas Rahn	thomas.rahn0303@gmail.com
John Di Girolamo	enerjohnzo91@gmail.com
David Siekut	davidsiekut@gmail.com
Dror Ozgaon	dror.ozgaon@gmail.com
Michael Lavoie	lavoie6453@gmail.com
Kwok-Chak Wan	martinwan1992@hotmail.com
Quang Tran	tran.quang@live.com

Project Summary

The objective of this project is to teach English through the use of space repetition and television series. The project will consist of a website that will enable users to watch a short segment of a television series. Afterwards, the script of the same segment will appear and will allow the user to highlight the words that he or she does not understand. These words will be explained to the user for the context of the segment they have just watched and they will be continuously quizzed about those words which will make up the space repetition aspect of the product.

Velocity

Sprint 1

In sprint 1, the team mainly focused on figuring out the actual flow of the application and come up with a database model. The team managed to complete a total of 38 story points at the end of the sprint.

Sprint 2

In sprint 2, two of the core features were implemented. These include being able to select a video as a user and being able to create, read, update, and delete videos as an administrator. The team managed to complete a total of 33 story points at the end of the sprint.

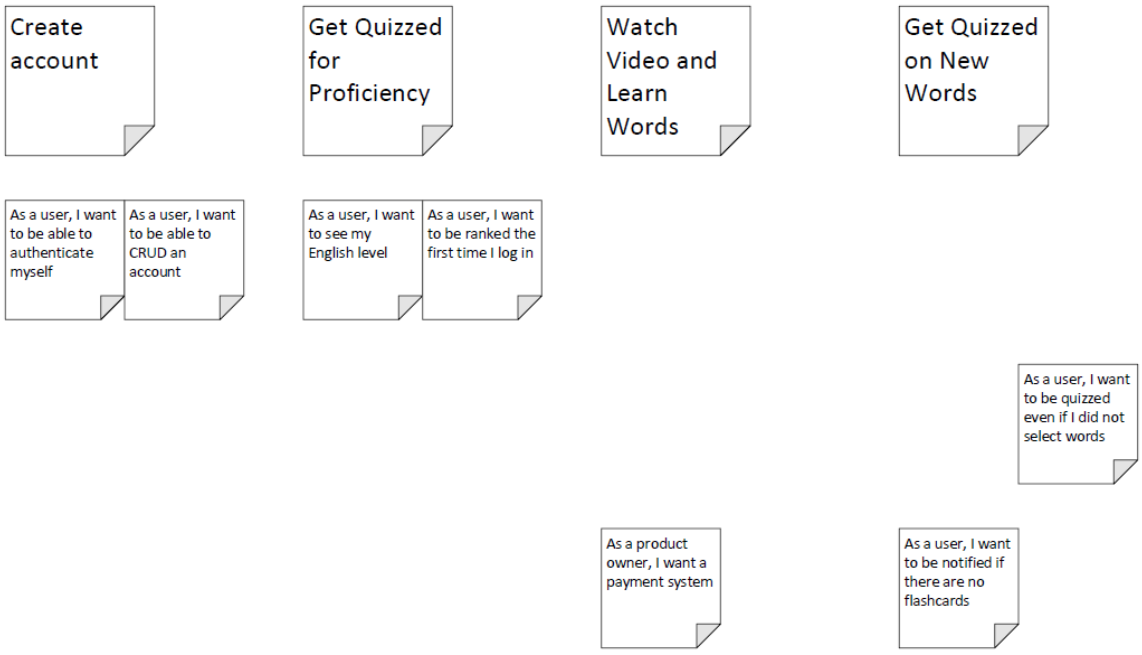
Sprint 3

In sprint 3, two additional core features were implemented. Namely, the flashcard and quizzing systems. The team managed to complete 28 story points at the end of the sprint.

Overall

(38+33+28)/3 = 33 ± 5 Story Points (15% error margin)

Story Map for Next Release



Overall Architecture and Database Model

The architecture for the LanguageLeap application is a slight variation of the MVC architecture deployed by Laravel (php framework for web development.) In the traditional Model-View-Controller architecture, the user interacts with the View which is handled by a Controller. Afterwards, the Controller modifies the Model which in turn updates the View.

In our variation, the Controller has the responsibility of updating the View instead of the Model. This can be seen by the diagram below:

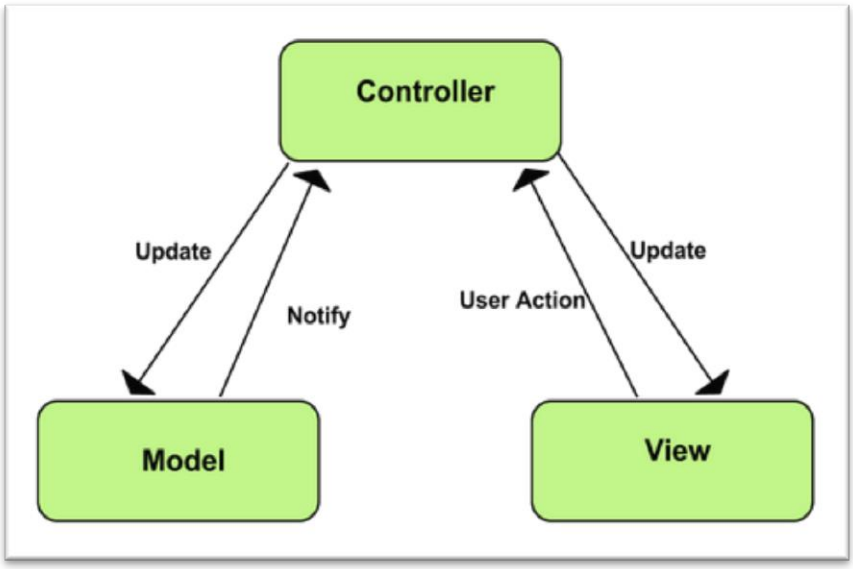


Figure-1: LanguageLeap variation of the Model-View-Controller architecture.
https://developer.chrome.com/static/images/mvc.png

Since LanguageLeap is a web application, it is more common to display the Database model rather than a class diagram. The Database model can be seen below.

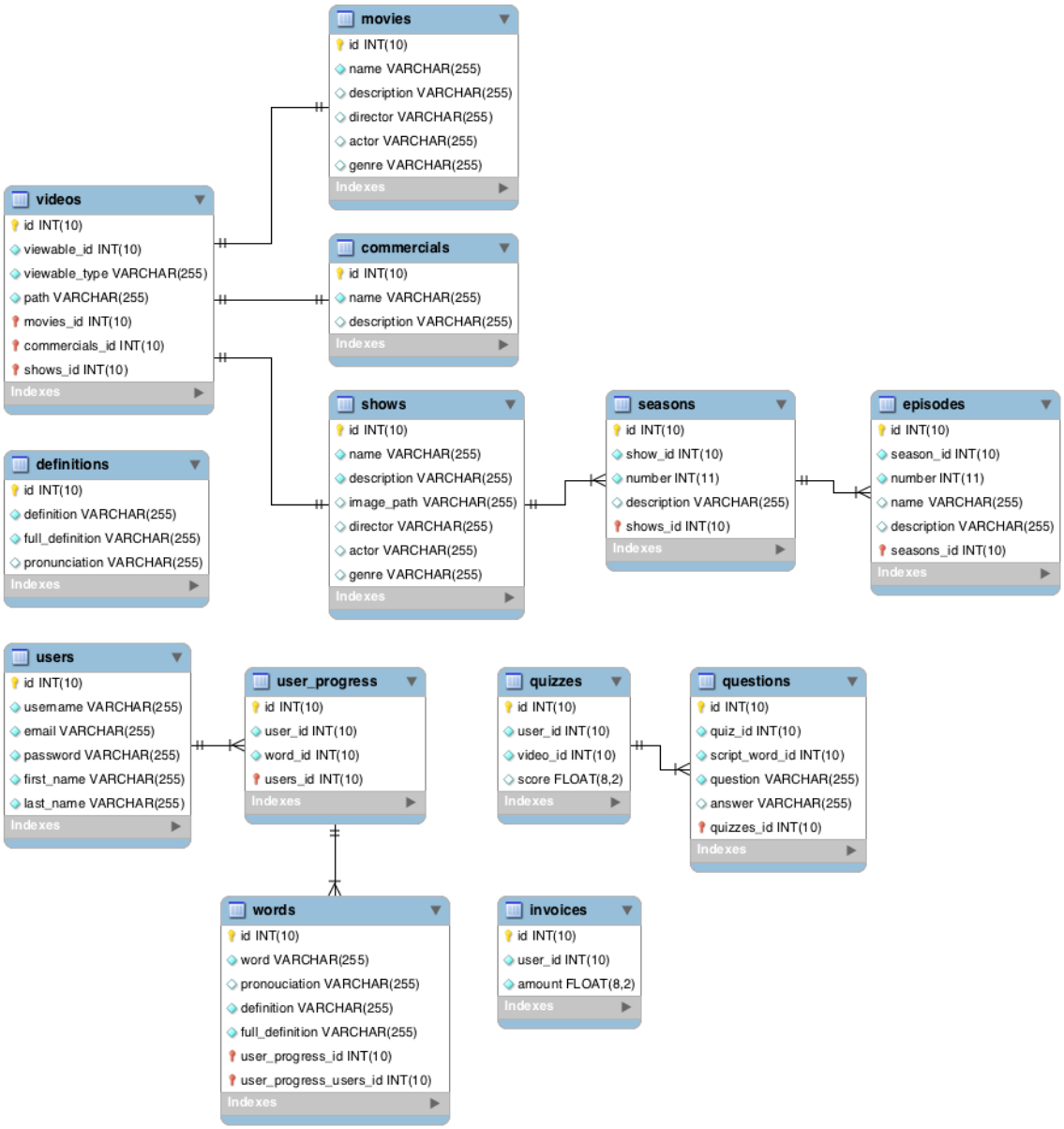


Figure-2: LanguageLeap Database model.

Completed Stories Summaries

See attached Acceptance Test Document.

Continuous Integration Strategy

For the purposes of our project, we maintain a continuous integration setup which is tied into the Github repository. We use Travis-CI's continuous-integration-as-service system as our primary CI system. We also have a secondary system using Bamboo that is locally hosted. Both systems are notified of new pushes onto the repository via Github webhooks integration.

When a developer pushes their commits onto the repository, Github notifies our CI setup, which then clones the affected branch and automatically runs our test suite.

Our test suite consists of unit and integration test cases, written using PHPUnit.

Our tests are primarily automated based on repository activity. With Bamboo, we can manually invoke a test run on any arbitrary branch. With Travis-CI, we can re-run a test run on an existing branch.

Relevant Development Links

Repository
https://github.com/alanly/languageleap

Project Tracking
http://jira.stumpfwerk.com

Team Communication
https://languageleap.slack.com

Continuous Integration
http://bamboo.stumpfwerk.com
https://magnum.travis-ci.com/alanly/languageleap